Risk Management Colloquium III

Integrated Life Cycle Risk Management

September 17-19, 2002

Sponsored by: NASA Office of Chief Engineer and Office of Safety and Mission Assurance

Hosted by: Ames Research Center

Located at: Hyatt Rickeys in Palo Alto, CA (5 miles from Ames Research Center)

Tuesday, September 17 **8:00 a.m. to 5:00 p.m.**

- ***Keynote Address***
  - *Bryan O'Connor, Associate Administrator for the Office of Safety and Mission Assurance*
  - *Theron M. Bradley, Jr., NASA Chief Engineer*
- *Program/Project Managers' Perspectives on Managing Risks*
- *Practitioners of Risk Management — Strategies and Approaches*
  - *Acquisition*
  - *Cost*
  - *Environmental*
  - *Export Control*
  - *Security*
  - *Health & Medical*
  - *Safety*
  - *Schedule*
  - *Technology Development*
- *Special Topic: Risk Management for Nuclear Systems*

Wednesday, September 18 **8:00 a.m. to 5:00 p.m.**

- *Independent Program Assessment Office (IPAO) Perspectives on Risk Management*
- *Systems Management Office (SMO) Perspectives on Risk Management*
- *Risk Management Training and Personnel Development*
- *International Partner Perspectives on Risk*
- *The Future of Risk Management Technology*
- *Expert Panel: "Integrated Life Cycle Risk Management"*

Thursday, September 19 **8:00 a.m. to 5:00 p.m.**

- *Risk Management — Safety and Mission Assurance Progress Report from Centers*
- *Tutorials*
- *Concluding Remarks and Wrap-up*

Open to NASA Personnel, NASA Contractors, and invited participants

---

For more information, visit the RMC III web site at http://risk.arc.nasa.gov/rmc3

Register Online — Hotel Reservations for the government rate are due by 8/26/02

Software Risk Management
(An evolving process)

September 18, 2002

Burton C. Sigal
Mission Assurance Office
Office of Safety & Mission Success
Jet Propulsion Laboratory

The Challenge

- The amount of flight software being flown and the complexity of demands on that software and on the changing approaches to its development are increasing dramatically, so it is becoming increasingly more important to...
- "...Do the right things right the 1st time..."
- Easy to say, but
  - How do we determine what are the 'right' set of assurance activities for a specific project?
  - What are the benefits of applying any set of assurance activities?
  - What are the residual risks associated with any selected set of assurance activities?
  - Is there an alternative set of assurance activities that is even better, e.g., less risk and/or lower cost?

Residual Risk Issues

- **What are the implications of the residual risks, if projects chose not to do individual assurance activities?**
  - If an assurance activity is not done, what can/has gone wrong?
  - If an assurance activity is used correctly, what problems/risks should be avoidable and what are the benefits?
  - If I don't choose or have funds to do specific assurance activities, what risks are being accepted by the project?
  - Are there redundancies in assurance activities with respect to individual risks?
  - Are there (critical) risks that have insufficient coverage?
  - Given a limited budget and specific project resource drivers, is the project buying the best set ofassurance activities?

Assurance Optimization Goals

The selection of a set of assurance activities such that:

For a **given set of resources**
(time, budget, personnel, test beds, simulators, ...)
benefits are maximized

or

For a given set of objectives
(science return goals; on-time and in-budget development; 99+% expectation of successful landing)
costs are minimized.

Assurance Costs & Benefits

Assurance activities have **costs**:

- Requirements inspections take skilled people's time
- Test-what-you-fly takes high-fidelity testbeds
- Bounds checking requires analysis and test case development

Assurance activities have **benefits**:

- Requirements inspections may catch problems early, when it is inexpensive to fix them
- Test-what-you-fly may catch problems that would jeopardize the mission
- Bounds checking may decrease the frequency of switching into safe mode

What's Needed for Assurance Optimization

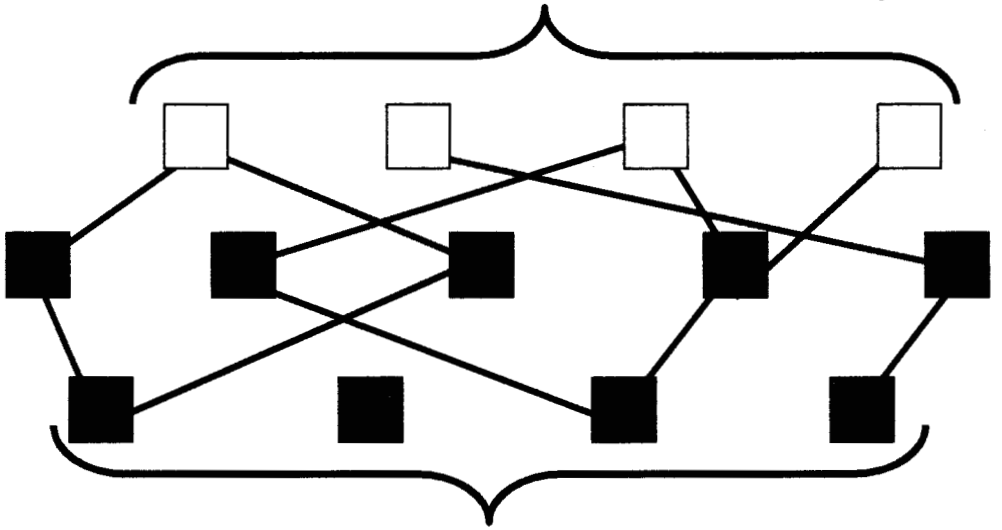
1. Models to calculate assurance **costs** & **benefits**-
   we use Defect Detection and Prevention (DDP)

2. Data to populate the model -
   We populate with metrics from experience (when available) augmented with experts' best estimates

3. Optimization over the model -
   We use Menzies' TAR2 treatment learning system (confirmed using simulated annealing)

DDP Cost/Benefit Model

Benefits = Σ attainment of requirements

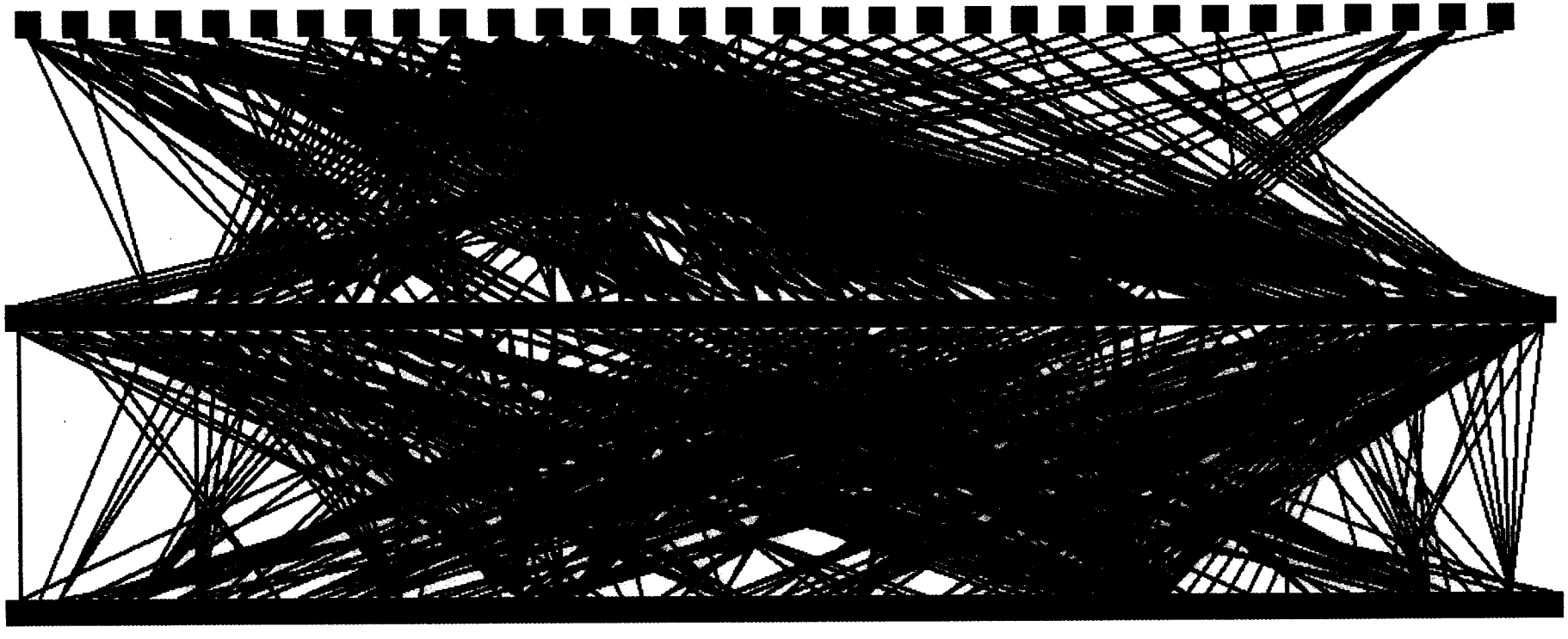


Costs = Σ costs of selected assurance activities

Model holds *quantitative* measures of:
How much each risk impacts each requirement, and
How much each assurance activity reduces each risk.

Risks are crucial intermediaries in the model
risks impact requirements to differing extents
assurance activities mitigate **risks** to differing extents

A DDP Dataset Populated from Real Experts



32 requirements, 69 risks, 99 assurance activities
352 non-zero quantitative requirement-risk links
440 non-zero quantitative assurance-risk links

A Typical Set of Project Software Risks

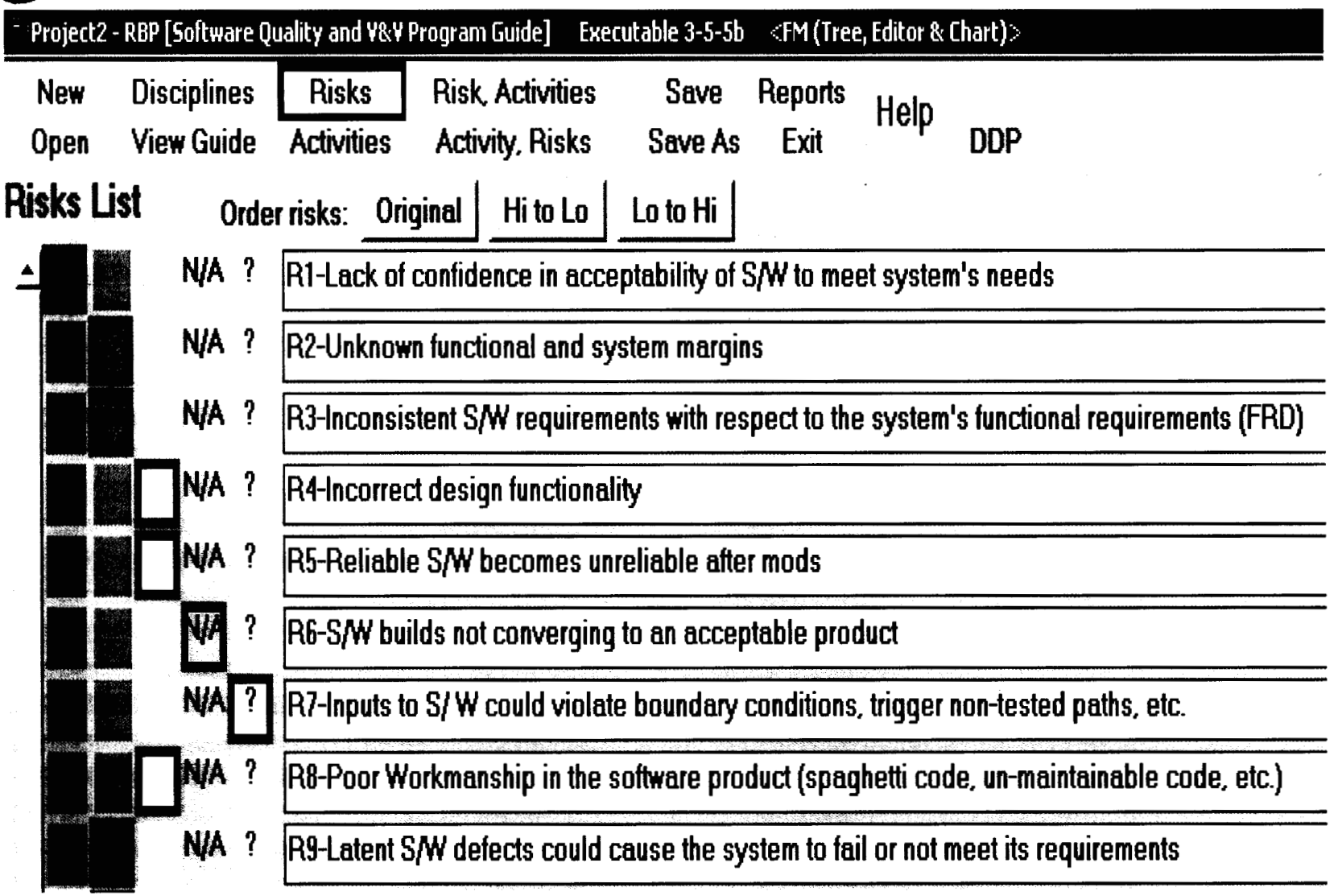
Project2 - RBP [Software Quality and V&V Program Guide] Executable 3-5-5b <FM (Tree, Editor & Chart)>

New Disciplines **Risks** Risk, Activities Save Reports Help
Open View Guide Activities Activity, Risks Save As Exit DDP

Risks List Order risks: Original | Hi to Lo | Lo to Hi

N/A	?	R1-Lack of confidence in acceptability of S/W to meet system's needs
N/A	?	R2-Unknown functional and system margins
N/A	?	R3-Inconsistent S/W requirements with respect to the system's functional requirements (FRD)
N/A	?	R4-Incorrect design functionality
N/A	?	R5-Reliable S/W becomes unreliable after mods
N/A	?	R6-S/W builds not converging to an acceptable product
N/A	?	R7-Inputs to S/ W could violate boundary conditions, trigger non-tested paths, etc.
N/A	?	R8-Poor Workmanship in the software product (spaghetti code, un-maintainable code, etc.)
N/A	?	R9-Latent S/W defects could cause the system to fail or not meet its requirements
N/A	?	R10-Late awareness (or lack of anticipation) of schedule, performance, cost and quality problems

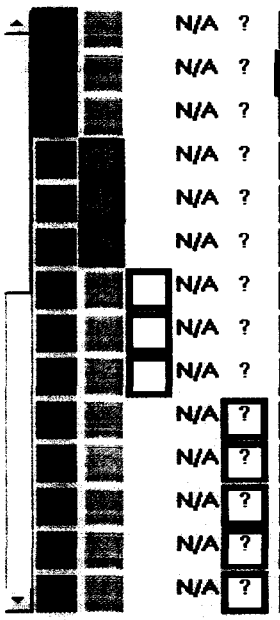
Initial Ranking of Project Software Risks



Risks Sorted By Weighting

New Disciplines **Risks** Risk, Activities Save Reports Help
Open View Guide Activities Activity, Risks Save As Exit

Risks List Order risks: **Original** | **Hi to Lo** | **Lo to Hi**



N/A	?	R1-Lack of confidence in acceptability of S/W to meet system's needs
N/A	?	R8-Poor Workmanship in the software product (spaghetti code, un-maintainable code, etc.)
N/A	?	R10-Late awareness (or lack of anticipation) of schedule, performance, cost and quality problems
N/A	?	R2-Unknown functional and system margins
N/A	?	R11-Software safety problem
N/A	?	R14-S/W fails in a harmful manner
N/A	?	R4-Incorrect design functionality
N/A	?	R6-S/W builds not converging to an acceptable product
N/A	?	R13-Lack of robustness of functions supported by S/W
N/A	?	R3-Inconsistent S/W requirements with respect to the system's functional requirements (FRD)
N/A	?	R5-No regression testing
N/A	?	R9-Latent S/W defects could cause the system to fail or not meet its requirements
N/A	?	R12-Executing faulty commands on a spacecraft
N/A	?	R15-H/W and system failures compounded by inappropriate S/W responses

Description of highlighted risk (read-only)

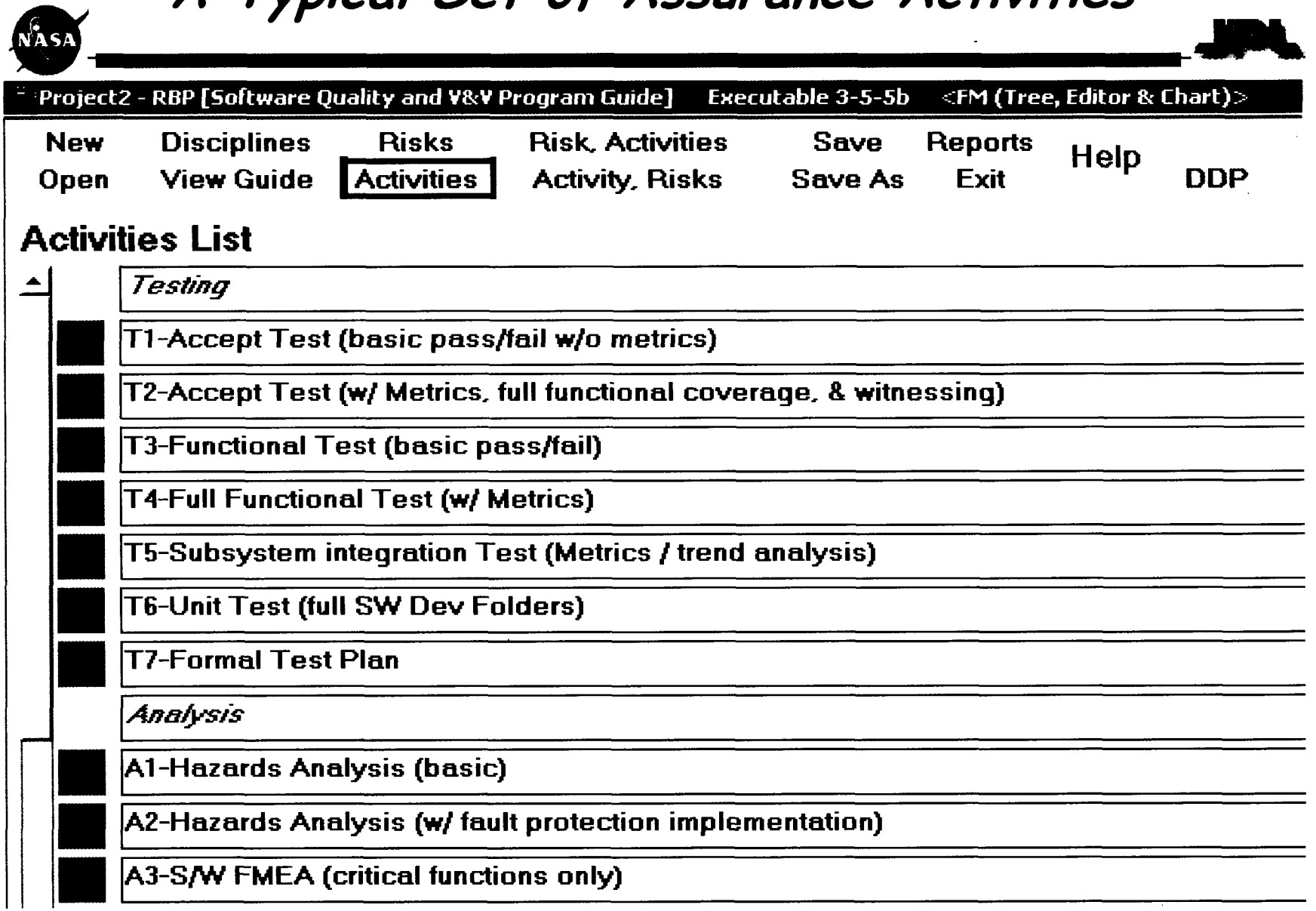
R8-Poor Workmanship in the software product (spaghetti code, un-maintainable code, etc.)
During the development process, code may become excessively complex because of highly coupled functional relationships, inadequate functional or object decomposition, or extensive and unanticipated requirements changes. Such code is often error-prone and difficult to maintain.

Notes of highlighted risk (click in box, then type to add and/or edit)

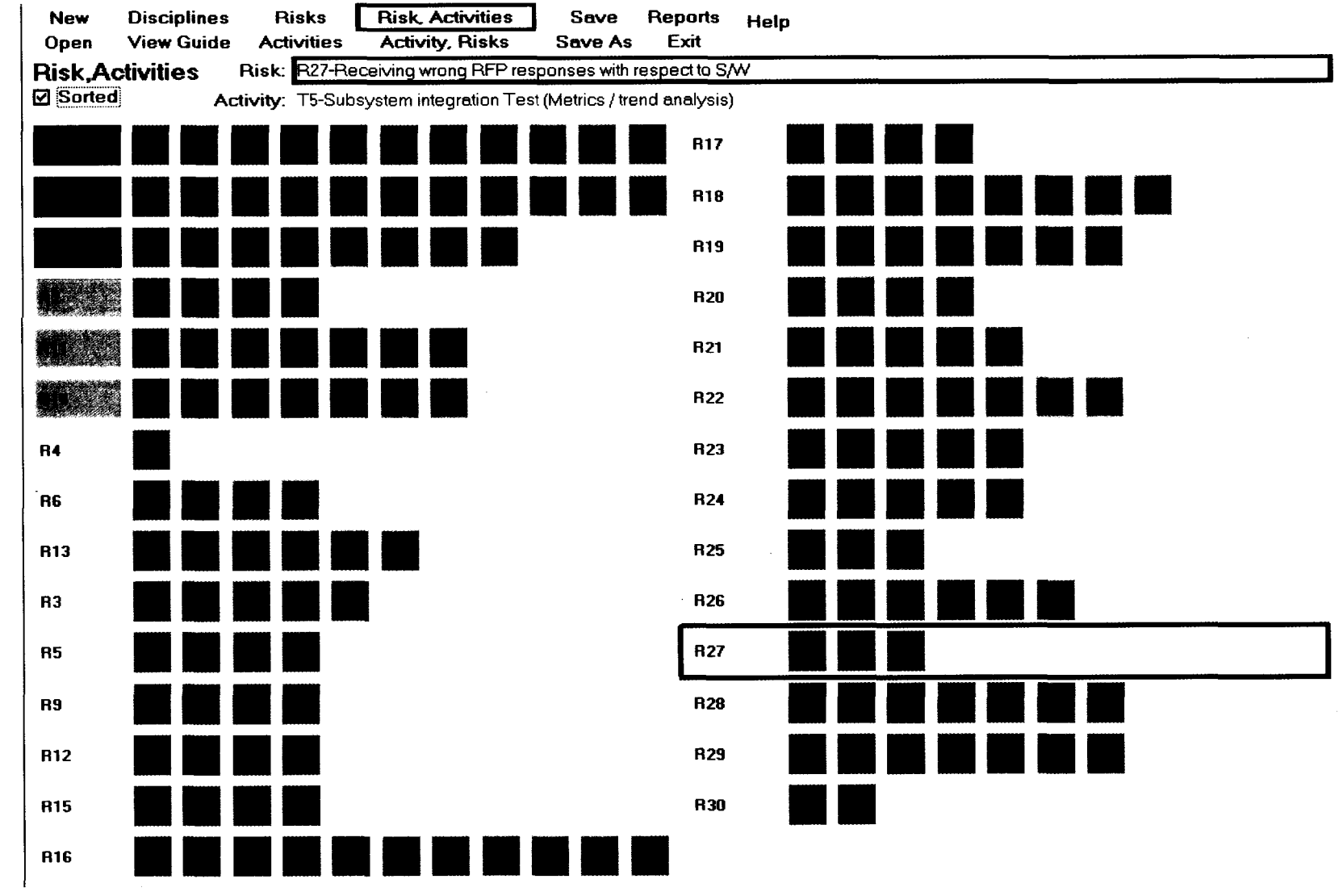
What do we know about past performance of developers/team?

Key to risk priority boxes ... *Low* N/A *Not Applicable* ? *Unknown*
= current priority; left-click box to set *= highlighted risk; left-click title to set*

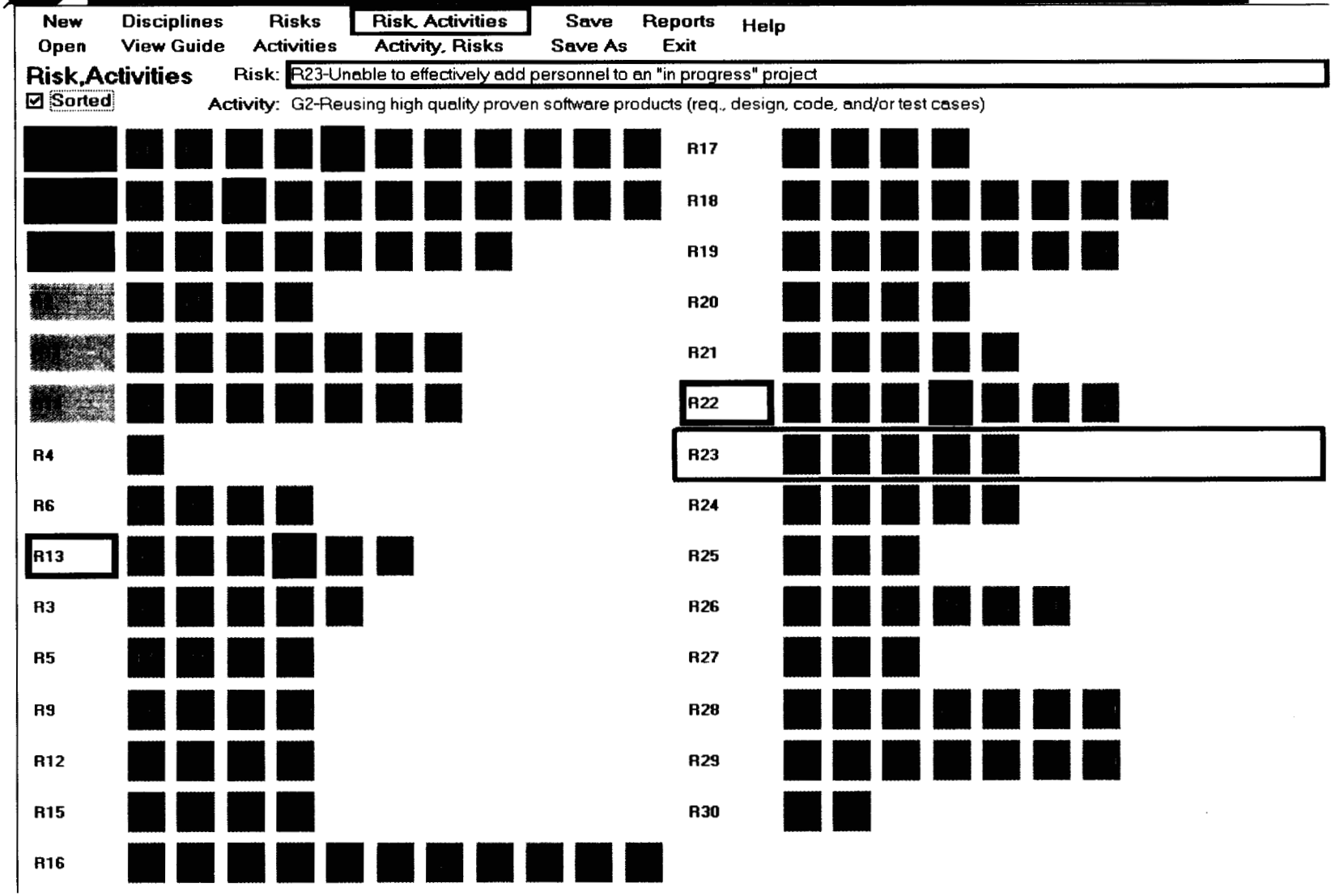
A Typical Set of Assurance Activities



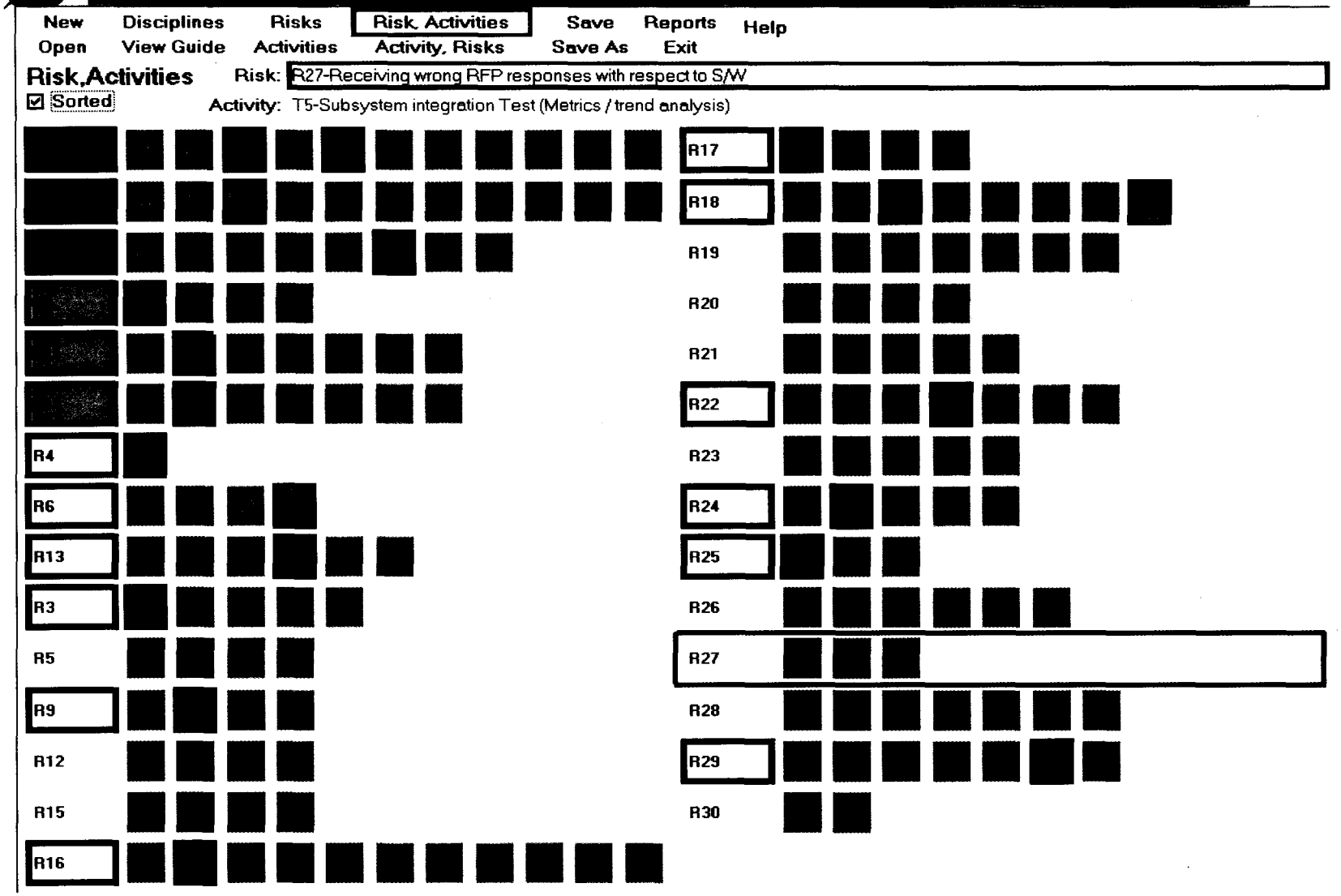
Assurance Activity Linked Risks



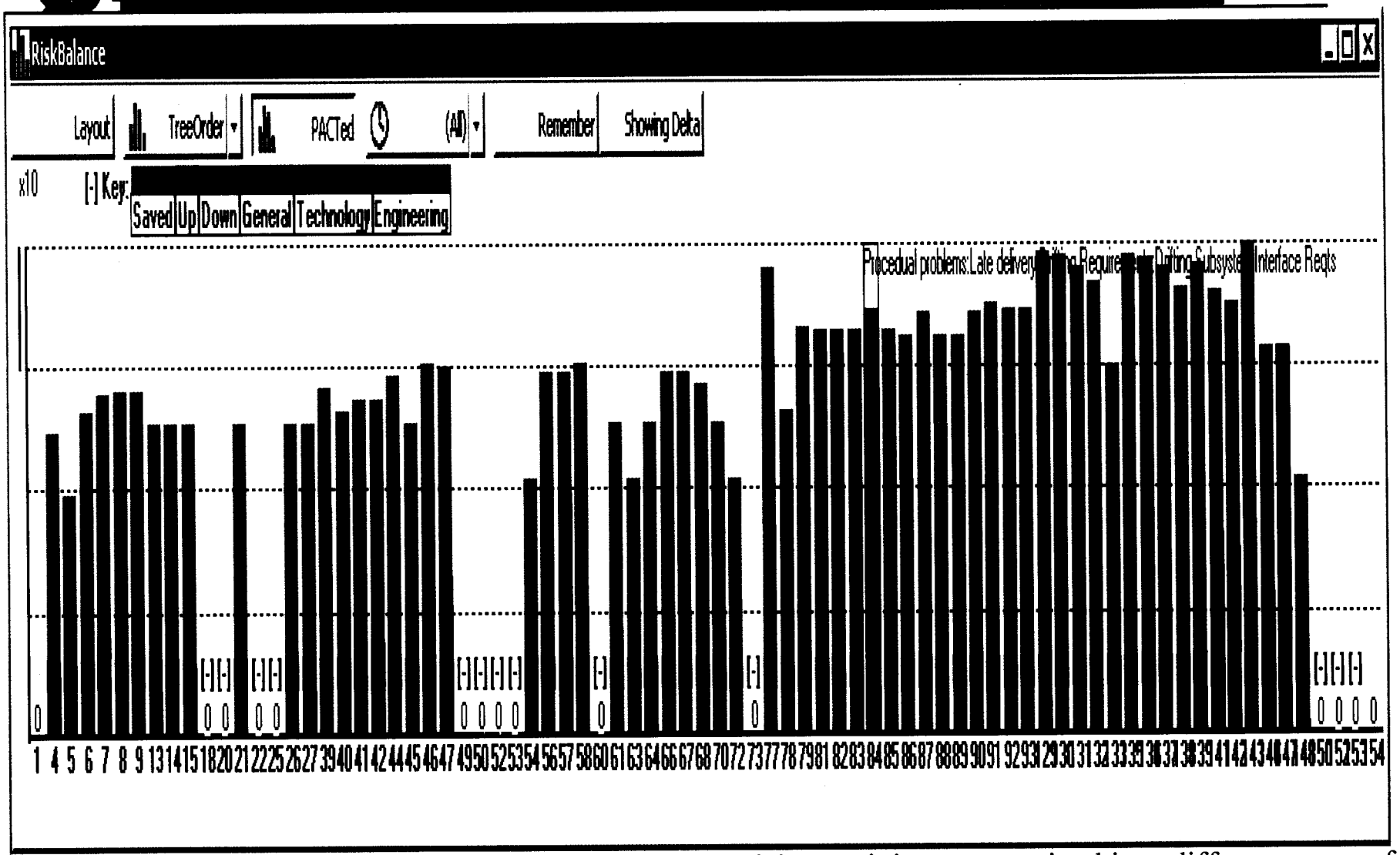
1st Cut At Assigning Assurance Activity



Final: Risks by Assurance Activities

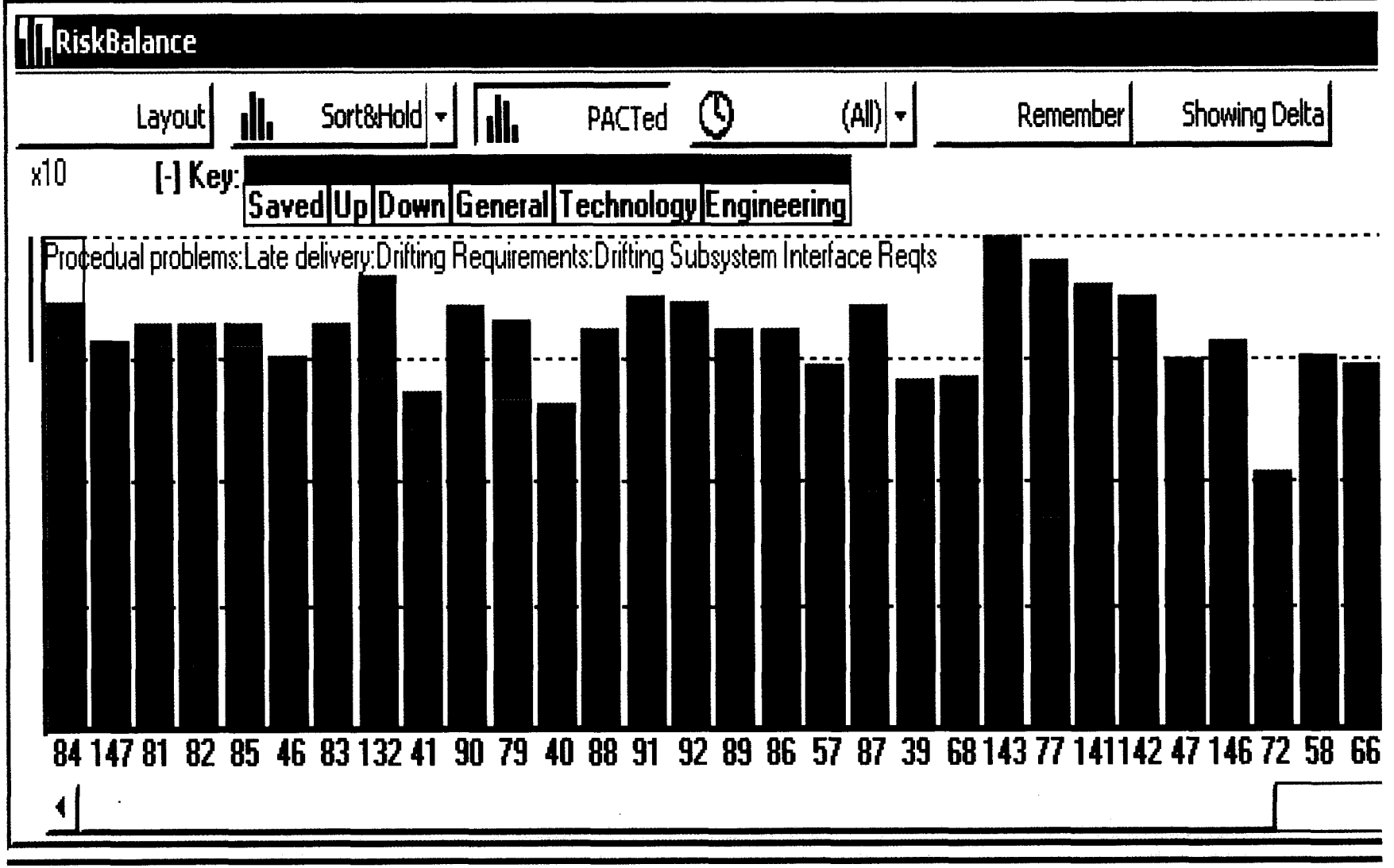


Risks Mitigated by Assurance Activities



Note: green = risk reduced; orange, red & purple = risk remaining, categorized into different areas of concern (specific to this particular study).

Pareto Sort by Risk

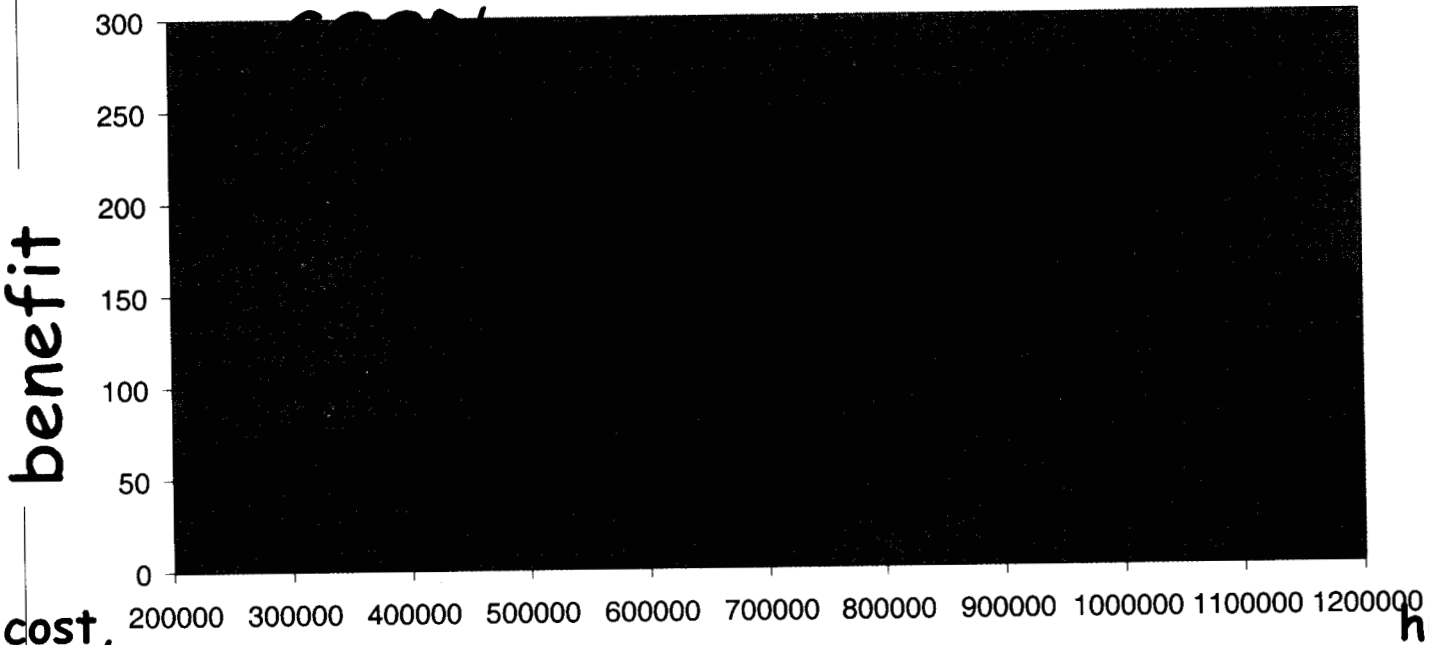


Note: green = risk reduced; orange, red & purple = risk remaining, categorized into different areas of concern (specific to this particular study).

Dataset before *Optimization*

low cost,
high benefit

high cost,
high benefit



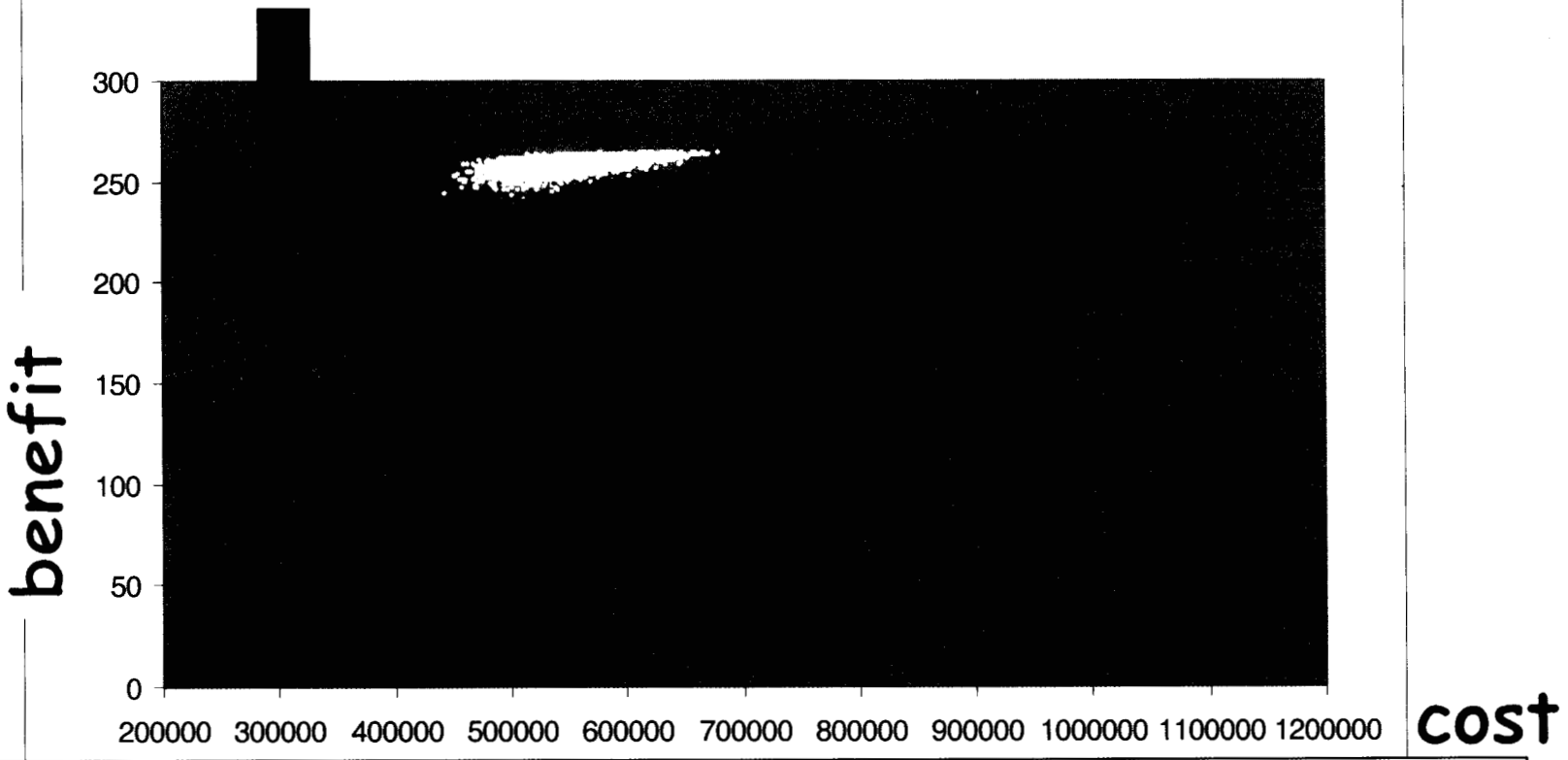
low cost,
low benefit

high cost,
low benefit

Each black point a randomly chosen selection of dataset's assurance activities. DDP used to calculate **cost** and **benefit** of each such selection.

Dataset after *Optimization*

Each white point is an optimized selection of dataset's assurance activities (33 critical ones are as directed by TAR2, other 66 chosen at random).



Menzies' TAR2 identified 33 most critical decisions:
21 of them assurance activities to perform
12 of them assurance activities to *not* perform.

Summary

- The amount of flight software being flown and the complexity of demands on that software and on the changing approaches to its development are increasing dramatically
- Meeting the quality demands of flight software requires new approaches to quality assurance optimization to ensure a robust product within project constraints
- Treating project specific risks as a resource to be traded like other project resources offers an effective solution
- Risk-assessment based tools which are easy to use over the project life cycle and allow tailoring, iteration, updating, and provide lessons learned, are a key part of that solution

Acknowledgements

Screenshots are taken from :
JPLer Steve Cornford's Defect Detection and Prevention (DDP) tool
and JPLer Tim Larson's Risk Balancing Profiles (RBP) tool

contributors (JPL)
Steve Cornford
Julia Dunphy
Martin Feather
Denise Howard
Chris Hartsough
John Kelly
Kelly Moran
Burt Sigal

contributors (other)
William Evanco (Drexel)
Steve Fickas (U. Oregon)
Richard Hutchinson (Wofford, SC)
Peter In (Texas A&M)
Jim Kiper (U. Miami, Ohio)
Tim Kurtz (NASA Glenn)
Tim Menzies (NASA IV&V)
Martha Wetherholt (NASA Code Q)

inspiration
Michael Greenfield (NASA Code Q), Tom Gindorf (JPL)

funding, management & guidance
Work sponsored by a combination of Software IV&V, Code R, and Code Q leveraged funding. This activity is managed locally at JPL through the Assurance and Technology Program Office (Chuck Barnes).